Information and Software Technology 55 (2013) 1260–1276

Contents lists available at SciVerse ScienceDirect

Information and Software Technology

journal homepage: www.elsevier.com/locate/infsof

Quality evaluation of floss projects: Application to ERP systems

Lerina Aversano, Maria Tortorella *

Department of Engineering, University of Sannio, Via Traiano, 82100 Benevento, Italy

ARTICLE INFO

Article history:
Received 20 May 2012
Received in revised form 25 January 2013
Accepted 27 January 2013
Available online 16 February 2013

Keywords:
Software evaluation
Open source
Enterprise resource planning
Software metrics

ABSTRACT

Context: The selection and adoption of open source software can significantly influence the competitiveness of organisations. Open source software solutions offer great opportunities for cost reduction and quality improvement, especially for small and medium enterprises that typically have to address major difficulties due to the limited resources available for selecting and adopting a new software system.
Objective: This paper aims to provide support for selecting the open source software that is most suitable to the specific needs of an enterprise from among the options offering equivalent or overlapping functionality.
Method: This paper proposes a framework for evaluating the quality and functionality of open source software systems. The name of the framework is EFFORT (Evaluation Framework for Free/Open souRce projecTs). It supports the evaluation of product quality, community trustworthiness and product attractiveness. The framework needs to be customised to the analysis of software systems for a specific context.
Results: The paper presents the customisation of EFFORT for evaluating Enterprise Resource Planning (ERP) open source software systems. The customised framework was applied to the evaluation and comparison of five ERP open source software systems. The results obtained permitted both the refinement of the measurement framework and the identification of the ERP open source software system that achieved the highest score for each chosen characteristic.
Conclusion: EFFORT is a useful tool for evaluating and selecting an open source software system. It may significantly reduce the amount of negotiation conducted among an enterprise's members and reduce the time and cost required for gathering and interpreting data. The EFFORT framework also considers the users' opinions by introducing relevance markers associated with the metrics and questions in the data aggregation process.

1. Introduction

Open source software has achieved more and more attention in the last few years. Many large and small enterprises are taking an interest in this growing software area. Indeed, significant benefits can be derived from open source software (OSS) – solutions, because they offer great opportunities for cost reduction, quality and efficiency improvement [20,21,27]. The adoption of OSS systems can appear an easy solution, because the software systems are free and can be downloaded from the Internet and installed or customised as needed. Organisations interested in reducing the licensing fees of proprietary software and avoiding the penalties and legal liabilities associated with their illegal use can consider open source software a plausible alternative [39]. However, less obvious than cost savings but equally important are the barriers to adopting OSS systems, such as their adoption costs and quality. Before adopting an OSS system, it is necessary that it and its business opportunities are understood and knowledge is gained regarding the advantages and constraints associated with its use. It is also necessary to understand the real needs of enterprises and select the OSS solutions that best satisfies those needs from among all the systems available that provide equivalent or overlapping functionality. This requires methods and tools to be applied to evaluation of the quality of OSS solutions and selecting the one that is best suited to the specific needs [14].

A widely held assumption in software engineering is that the external quality characteristics of a software system are correlated to its internal quality characteristics and thus that source code metrics provide useful data for assessing the quality of a software system [9,30–32,34]. The International Organization for Standardization (ISO) published for the first time in 1991 the ISO/IEC 9126 standard [30–32], which defines a quality model for software products to be considered as a reference for their evaluation. Nevertheless, the original standard ISO/IEC 9126 and its later versions are inadequate for characterising the quality of a Free *libre* Open Source Software (F*l*OSS) project [14,39], because a F*l*OSS project is different from a closed source one in terms of production, distribution and support modalities, which are more relevant than product-related characteristics. Open source software is in most cases developed in a very open environment, and several sources

* Corresponding author. Tel.: +39 0824 305554; fax: +39 0824 50552.
E-mail addresses: aversano@unisannio.it (L. Aversano), tortorella@unisannio.it (M. Tortorella).

0950-5849/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.infsof.2013.01.007

of information that would not be available in a proprietary software system can be used in its evaluation [50]. Specifically, open source software allows examination of the source code, performance of white box testing, and analysis. In most open source projects, it is also possible to obtain access to the version control system, mailing lists and bug management databases. This additional information allows prospective users to conduct a more comprehensive evaluation of both the software system and the project using it. Therefore, while traditional software evaluation methods mostly focus on the software itself, its functionality, usability and price, a much broader approach can be adopted for evaluating OSS systems using additional available information.

The paper presents an advanced quality model for F*l*OSS projects. It extends the current methods by considering additional characteristics unique to F*l*OSS projects, such as the quality of the community and product attractiveness. Specifically, the paper proposes the framework EFFORT—Evaluation Framework for Free/Open souRce projecTs—that provides guidelines, procedures and metrics for applying the defined quality model for evaluating F*l*OSS projects. The aim is to support the comparison of a set of F*l*OSS software projects and select the system that is best suited to the needs of a potential adopter [1]. Among the entities that may be interested in adopting a F*l*OSS system are an enterprise, a developer that wants to enhance a F*l*OSS system, and a potential sponsor that wants to distribute such a system. Each of these entities considers some of the various quality aspects that are included in the proposed framework. In addition, the framework is parametric with respect to the evaluated characteristics and permits its customisation for modelling the specific needs of an evaluator.

To achieve a more detailed evaluation, the EFFORT framework is defined in a generic manner and can be instantiated for a specific context before being applied to evaluation of a specific set of F*l*OSS projects. Knowledge of the application domain and characteristics of a software system are very important to improving and extending the quality measurement framework to be applied to evaluation of a software system [2,3]. Customisation requires the extension of EFFORT by considering additional specific aspects that capture the special features of the application domain. This paper describes the customisation of the framework to the context of Enterprise Resource Planning (ERP) systems and applies the customised framework to the evaluation of five F*l*OSS ERP projects. Indeed, F*l*OSS ERP projects represent important examples of how an organisation can achieve competitive advantages by exploiting the opportunities provided by OSS solutions. In addition, the selection and adoption of one of the options cannot be addressed in a superficial manner.

The remainder of this paper is organised as follows. Section 2 analyses existing models and tools for evaluating and selecting F*l*OSS projects. It also summarises related studies concerning the evaluation of ERP software system quality. Section 3 provides a description of the EFFORT framework. Section 4 describes the customisation of the EFFORT framework to the evaluation of ERP F*l*OSS projects and evaluates five ERP F*l*OSS projects. Concluding remarks are given in the last section.

2. Related works

This section discusses the existing literature, with reference to (i) studies aimed at evaluating the quality of OSS systems and (ii) approaches to the selection and comparison of ERP systems.

2.1. Evaluation of software system quality

The quality evaluation of software systems is a relevant issue for software engineers, and various models have been proposed in the literature [6,35]. The traditional models are the McCall and Boehm's models [9,10,34,37], the more widely accepted ISO/IEC 9126 model [30–32], and its more recent implementation by SQuaRE ISO 25000:2005 [32]. These models decompose the quality concept into a hierarchy of criteria and attributes further decomposed into lowest-level metrics [22].

The issue of quality evaluation is frequently related to OSS systems. However, due to the nature of OSS development, according to which standard practices include open access to the source code, shared artefacts, peer review of code, asynchronous global development and lack of formal supports, traditional models may not be sufficient. As a consequence, quality models specifically focused on OSS development have been proposed in the literature.

The Open Source Maturity Model (OSMM) assumes that the quality of an open source project depends on its maturity [26] and decomposes it into six elements: product software, support, documentation, training, product integration and professional services. Although OSMM is simple to apply, it does not take into account some important software artefacts, such as the source code itself.

Kamseu and Habra analysed the different factors that could influence the adoption of an OSS system [33]. They considered three dimensions of the quality of an open source project: the development process, including the management and support processes; community, including developers and contributors; and product, concerning the product after its release.

Sung et al. focused on the quality of the product by adapting the ISO/IEC 9126 standard to F*l*OSS products [48]. The authors identified some problems with the evaluation of an OSS product using this standard, such as the difficulty of collecting information when the developers do not make it public.

Another proposed methodology for assessing F*l*OSS projects is Qualification and Selection of Open Source software (QSOS) [12]. The QSOS method was developed to make reusable evaluations. Each OSS project is evaluated by applying the evaluation criteria and assigning an absolute score, weight and threshold to each criterion. QSOS provides a tree hierarchy of evaluation criteria with scoring for each leaf criterion. Criticisms of this methodology have been made with reference to the small scoring range [19]. Moreover, because there is no definition regarding the first two levels of criteria, there is ambiguity in its application.

The OpenBRR project–Business Readiness Rating for Open Source [41]–was developed for the same purpose of QSOS. The approach includes a pre-screening phase that ends with a few viable candidates for selection. An evaluation template is tailored by reviewing and selecting the appropriate evaluation criteria from a proposed hierarchy. A weight is assigned to each metric to be used for aggregating the metric scores and obtaining the score of each category.

QualiPSo—Quality Platform for Open Source Software—is one of the largest initiatives of the European Union. It defines an evaluation framework for the trustworthiness of F*l*OSS projects [15–17]. The trustworthiness is defined in terms of product quality and considers as-is utility, exploitability in development, functionality, interoperability, reliability, performance, security, cost-effectiveness, customer satisfaction and developer quality. An assessment model has been defined that makes use of the results of an analysis of information about the trustworthiness characteristics available in OSS portals and repositories using the OP^{2}A (Open source Product Portal Assessment).

Wheeler defined a F*l*OSS selection process called IRCA—Identify Read reviews Compare and Analyse [50]. It considers in its evaluation important attributes such as functionality, cost, market share, support, maintenance, reliability, performance, scalability, usability, security, flexibility/customisability, interoperability, and

legal/license issues. Unfortunately, the method does not define metrics for doing the evaluation.

The SQO-OSS—Software Quality Observatory for Open Source Software—model was constructed to support an automated software evaluation system [44,46]. Its variables are mainly metric-oriented, and human intervention is minimal. The model evaluates the main aspects of OSS development, such as source code and community. The SQO-OSS model does not evaluate functionality; rather it focuses on aspects of OSS quality, such as OSS project maintainability, reliability and security.

A comparison of the models described above was conducted. The first aim was to identify the models that were most complaint with the ISO/IEC 9126 standard, analysing the coverage and features they had in common. The analysis focused on the most interesting characteristics of the open source software to identify the features investigated in the models.

Fig. 1 and Table 1 show the results of the analysis. They also show how EFFORT compares with the other quality models, but this aspect will be discussed later. Table 1 shows the results of the first analysis. It should be noted that a standard characteristic was considered as covered by a model if it took into account at least one of its attributes. Table 1 shows that not all the models considered take into account the ISO standard quality characteristics. The highest coverage is exhibited by IRCA, but it does not provide an adequate operational tool for its application. The table also shows that the in-use quality is the least-considered quality characteristic. This is due to the difficulty of objectively measuring the metrics related to in-use quality, because they depend greatly on the user.

Fig. 1 graphically shows the overlap of the models without considering the standard characteristics. The used shapes do not have are not related to the completeness of the approaches. Indeed, the figure only regards the following subset of macro characteristics of the models that are considered relevant for assessing FlOSS systems: licensing, documentation, support, adoption, community, maturity, architecture and cost-effectiveness. The figure does not include the SQO-OSS and QualOSS models because they do not analyse the characteristics considered.

Table 1 and Fig. 1 show that some of the models, such as QualiPSo and the Sung–Kim–Rhew model, mostly emphasise product-intrinsic characteristics and only slightly consider the other OSS dimensions. Such models provide major coverage of the ISO standard. Other models that consider the OSS aspects offer reduced coverage of the ISO quality attributes. IRCA is the only exception: it is sufficiently complete, but it is a conceptual model; it does not provide an evaluation framework, and it is not formalised.

2.2. Evaluation of ERP software systems

Several research studies on the evaluation of ERP software systems can be found in the literature, confirming a growing interest in this topic. They describe research undertaken to compare the advantages and disadvantages of such systems.

Numerous approaches have been taken in evaluating ERP systems. They typically are approaches used for the evaluation of commercial systems, and only a few of them specifically pertain to FlOSS ERPs.

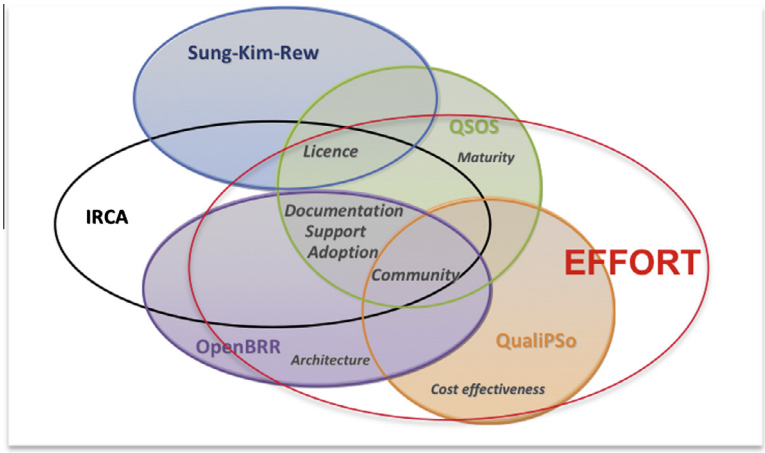

Fig. 1. Comparing the open source software quality evaluation models.

Table 1
Comparison among the proposed quality models with reference to the ISO standard.

ISO/IEC 9126	Quality models							
	SQO-OSS	Sung-Kim-Rhew	IRCA	QSOS	OpenBRR	QualOSS	QualiPSo	EFFORT
Functionality		Functionality	Functionality	Functional adequacy			Functionality	Functionality
Reliability	Reliability		Reliability	Maturity, Quality Assurance	Security		Reliability, Developer quality	Reliability
Usability		Usability	Usability	Exploitability	Usability			Usability
Efficiency			Performance		Performance		Performance	Efficiency
Maintainability	Maintainability		Maintainability/ Longevity	Modularity, Documentation	Documentation			Maintainability
Portability		Portability	Interoperability	Packaging			Interoperability	Portability
In use quality					Security		As-is utility, Customer satisfy.	

An approach to identifying and grouping the main criteria for selecting an ERP system, based on a set of aspects of the software system to be investigated, is proposed in [8].

Evaluation-Matrix (http://evaluation-matrix.com) is a platform for comparing management software systems. The approach has two main goals: to construct a list of characteristics representing the most common needs of a user and to have at the user's disposal a tool for evaluating available management software systems.

In Ref. [23], Nah schematically summarises the pros and cons of adopting an ERP system for Small and Medium Enterprises (SMEs). Such enterprises have few resources that can be dedicated to selection, acquisition, configuration and customisation of an ERP system. Moreover, ERPs are generally designed to fit the needs of large companies. Adopting a FlOSS ERP could partially fill this gap. It emerges from the literature that the adoption of FlOSS ERPs is more advantageous for SMEs [28,50]. As an example, some of the advantages arise from the possibility of trying the system (not just using a demo), reduction of vendor lock-in, a low license cost and the possibility of in-depth personalisation.

Reuther and Chattopadhyay performed a study to identify the main critical factors for selecting and implementing an ERP system to be adopted by an SME [42]. The factors identified were grouped in the following categories: technical/functional requirements, business drivers, cost drivers, flexibility, scalability, and other factors specific to the application domain. This research was extended by considering the context of F/OSS projects [43,52].

Wei, Chien and Wang defined a framework for selecting an ERP system based on the Analytic Hierarchy Process (AHP) technique [51]. This is a technique for supporting multiple-criteria decision problems that suggests how to prioritise a set of alternatives and importance of the relative attributes.

Bernroider and Stix [7] defined an approach that combines the merits of two prominent concepts individually applied in decision-making: the Utility Ranking Method (URM) and the Data Envelopment Analysis (DEA). In addition, the method calculates the distances between the desired system profile, defined in the additive multi-attribute utility model, and the individual alternative profiles, calculated by the DEA-derived optimisation process.

Liao et al. [36] presented a model based on linguistic information processing for dealing with such a problem. In [24,25], a method is proposed for selecting ERP systems based on fuzzy-data envelopment analysis. The model was used to evaluate each ERP software solution using subjective judgments made by a group of highly prestigious IT experts in the petrochemical industry. ERP tool selection has also been addressed in [11], in which the authors proposed a structured decision-making selection process for ERP tools. They applied a fuzzy cognitive map offering an organised and structural outline for the acquisition of an ERP tool.

Open Source ERP Guru is a platform that can support users interested in comparing open source ERP solutions [40]. The platform provides on-line demo of the most relevant open source ERP systems to allow easy comparisons by interested users. The results of a comparison depend on the user evaluation, and the platform does not provide a comparison matrix. The criteria suggested for the evaluation are the following: customisation complexity, functionality, user experience, development activity, knowledge, production-readiness, integration, migration, Total Cost of Ownership (TCO) [34], and community.

The models analysed are quite different, but they share a common goal of identifying critical factors for the selection of ERP systems. Table 2 includes a comparison of the models and identifies common elements. The rows of the matrix in Table 2 contain aspects considered in at least one model, while the columns refer to the models themselves. The presence of a tick in cell *i*, *j* means that factor *i* is totally or partially covered by model *j*. The criteria considered by the greatest number of models pertained to functionality, usability and costs, followed by support services, system reliability and customisability. One can observe from Table 2 that the Birdogan and Kemal model is the most complete, although it does not provide an operational approach for evaluating the characteristics listed above. The quality model that more completely considers the evaluation of the open source nature of ERP systems is OS ERP Guru. However, it should be noted that OS ERP Guru provides a high-level list of the characteristics to consider but does not provide any technical details concerning how to measure the characteristics. OS ERP Guru also does not include evaluation of the technical quality of ERP software products.

Table 2
Comparison of the ERP evaluation models proposed in literature: common elements.

Criteria	Model						
	Birdogan Kemal	EvaluationMatrix	Wei Chien Whang	Reuther Chattopadhyay	Zirawani Salihin Habibollah	Open Source ERP Guru	EFFORT
Functionality	√	√	√	√		√	√
Usability	√	√	√		√	√	√
Costs	√	√	√	√		√	√
Support Services	√	√	√		√		
Vendor's vision	√						
System reliability	√		√	√		√	√
Interoperability	√						√
Market share	√	√	√				
Domain knowledge of providers	√					√	√
References and reputation of vendors	√		√				√
Partnership	√						
Integration/Modularity	√				√	√	√
Implementation time	√		√				√
Software methodology	√						
Consulting	√						
Customization and flexibility		√	√	√	√		√
Migration		√	√		√		√
Technical quality	√	√		√			√
Develop activity			√				√
Community			√				√
Business competitive advantage				√			

Table 2 also includes the criteria that were considered in customising the EFFORT quality model. EFFORT was developed to overcome the limitation of the existing models and propose a practical method for gathering and aggregating metrics. This comparison of the proposed approaches with EFFORT will be discussed later.

3. Proposed approach

The evaluation framework, EFFORT, was developed to evaluate FlOSS systems and can be considered a base framework that can be customised to the specific context of the open source system to be selected. The development of EFFORT took into account a comparative analysis of the previously described approaches and identification of their deficiencies. Many of the approaches described analyse quality characteristics specific to Open Source Software systems, such as the quality of the community managing the system, and do not take into account the quality of the software product. On the contrary, in [18], the authors stated that the greater the system quality (software product) and service quality (community service support) are, the more the system is used. For this reason, two of the quality characteristics considered address *product quality* as well as *community quality* [4,49]. In addition, previous studies have shown that system quality influences the information system use and therefore its adoption [29,47]. In particular, those aspects that reflect the characteristic of the software product's attractiveness play an important role in the user's perception of the quality of the software system . The higher the perceived quality of a software product is, the higher the software product attractiveness can be considered. Thus, the third quality characteristic that EFFORT considers is the software *product attractiveness*.

In addition, according to the ISO Standard 8402 [30], quality is the totality of characteristics of an entity that bear on its ability to satisfy stated and implied needs. Therefore, even in the context of OSS, a software product is said to be of a good quality if it satisfies stated and implied users' needs. The users' needs change on the basis of the software system application domain. For this reason, EFFORT has to be instantiated to the specific application domain before being used to conduct a complete quality evaluation of specific open source software systems.

EFFORT was developed on the basis of the Goals Questions Metrics (GQM) paradigm [5]. This paradigm guides the definition of a metric program on the basis of three abstraction levels: the conceptual level, referred to as the definition of the *goals* to be achieved by the measurement activity; the operational level, consisting of a set of *questions* concerning the way the assessment or achievement of a specific goal is addressed; and the quantitative level, pertaining to identification of a set of *metrics* to be associated with each question.

To reiterate, the framework considers the quality of a FlOSS project as a synergy of the following three main quality characteristics:

- *Quality of the product* developed within the project. This is one of the main aspects indicating the project quality. It is unlikely that a product of high quality is developed within a low-quality project. Using the ISO/IEC 9126 standard, product quality is evaluated without considering the open source nature of the analysed software system.
- *Trustworthiness of the community* of developers and contributors. This indicates the degree of trust that a user has in a community with respect to the support offered. In fact, one of the main concerns of a potential user of an open source software system is the availability of support in case of problems. The adoption of an open source software system does not constrain the community to offer support. Support is provided in ways that differ greatly from one community to another.
- *Product attractiveness*. This characteristic considers all the factors that influence the adoption of a product by a potential user, who perceives convenience and usefulness in using it.

Based on the study of the literature and standards, the three quality characteristics were decomposed into the set of quality attributes shown in Fig. 2 and making up the EFFORT quality model. Thus, the EFFORT measurement framework includes three goals, each one corresponding to each first-level attribute. *Questions* map second-level characteristics. Considering its complexity and the number of aspects taken into account, *Goal 1* has been divided into sub-goals. The sub-goals concern the characteristics of the software product quality, as indicated in Fig. 2. For simplicity, the figure does not present the third level related to the questions

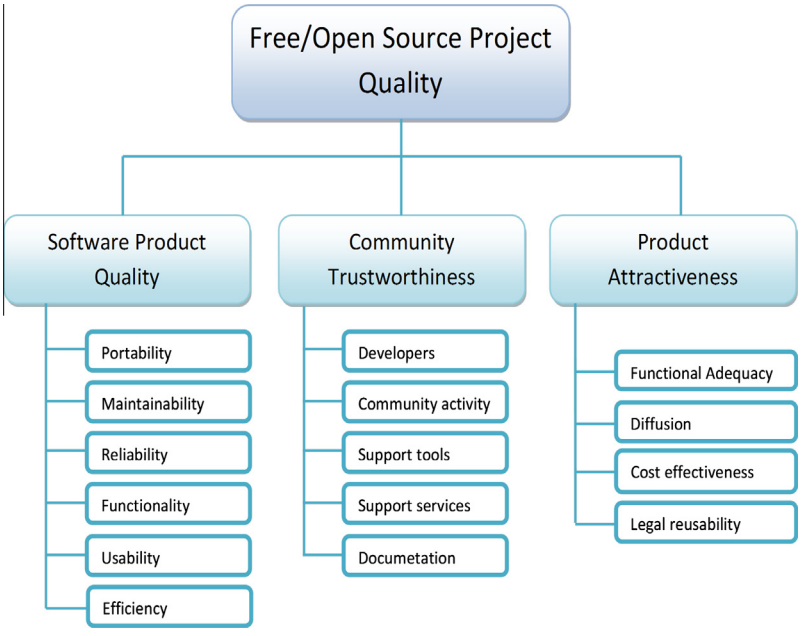

Fig. 2. Quality model defined by EFFORT and associated quality characteristics.

related to the software product quality sub-goals and metrics used in answering all the questions.

The following subsections describe the three goals and provide a formalisation of the goal itself, incidental definitions of specific terms and the related lists of questions. The questions listed in each subsection are answered through the evaluation of a set of associated metrics. This paper does not present all of the metrics considered, but several are presented completely. Questions and metrics associated with the goals are summarised in Tables 3–7. Each table includes the goal it addresses, the questions considered, identifiers and descriptions of the metrics, and the application and source of the data for evaluating the metrics.

Data were collected by analysing the documentation, trackers, repositories and official web sites of the projects. In addition, for

Table 3
Questions in the EFFORT measurement framework pertaining to product quality.

Goal 1 – Product quality	
Sub-goal 1a: *Analyse the software product with the aim of evaluating it with respect to portability from the software engineer's point of view*	
Q 1a.1	What degree of adaptability does the product offer?
Q 1a.2	What degree of installability does the product offer?
Q 1a.3	What degree of replaceability does the product offer?
Q 1a.4	What degree of coexistence does the product offer?
Sub-goal 1b: *Analyse the software product with the aim of evaluating it with respect to maintainability from the software engineer's point of view*	
Q 1b.1	What degree of analyzability does the product offer?
Q 1b.2	What degree of changeability does the product offer?
Q 1b.3	What degree of testability does the product offer?
Q 1b.4	What degree of technology concentration does the product offer?
Q 1b.5	What degree of stability does the product offer?
Sub-goal 1c: *Analyse the software product with the aim of evaluating it with respect to reliability from the software engineer's point of view*	
Q 1c.1	What degree of robustness does the product offer?
Q 1c.2	What degree of recoverability does the product offer?
Sub-goal 1d: *Analyse the software product with the aim of evaluating it with respect to functionality from the software engineer's point of view*	
Q 1d.1	What degree of functional adequacy does the product offer?
Q 1d.2	What degree of interoperability does the product offer?
Q 1d.3	What degree of functional accuracy does the product offer?
Sub-goal 1e: *Analyse the software product with the aim of evaluating it with respect to usability from the user's point of view*	
Q 1e.1	What degree of pleasantness does the product offer?
Q 1e.2	What degree of operability does the product offer?
Q 1e.3	What degree of understandability does the product offer?
Q 1e.4	What degree of learnability does the product offer?
Sub-goal 1f: *Analyse the software product with the aim of evaluating it with respect to efficiency from the software engineer's point of view*	
Q 1f.1	How the product is characterised in terms of time behaviour?
Q 1f.2	How the product is characterised in terms of resources utilisation?

Table 4
Questions and metrics in the EFFORT measurement framework pertaining to sub-goal 1a.

SUB-GOAL 1a – Portability				
Question		Metric		
Id	Name	Id	Name	Application and data source
Q 1a.1	What degree of adaptability does the product offer?	M 1a.1.1	Number of operating systems supported	Analysis of the documentation on the official web site
Q 1a.2	What degree of installability does the product offer?	M 1a.2.1	Time required for installation	Measurement of the required for installation of the product
		M 1a.2.2	Availability of the installation manual	Analysis of the documentation on the official web site
		M 1a.2.3	Automation level and use of installation scripts	Evaluation of the percentage of steps that can be automated during the installation
		M 1a.2.4	Dependence on third-party components	Evaluation during the product installation
		M 1a.2.5	Nominal length of the installation procedures	Analysis of the installation manual
		M 1a.2.6	Number of configuration files	Analysis of the documentation on the official web site
		M 1a.2.7	Availability of default options	Evaluation during the product installation
		M 1a.2.8	Internationalisation of the manual	Analysis of the documentation on the official web site
		M 1a.2.9	Number of unforeseen issues	Evaluation during the product installation
		M 1a.2.10	Degree of knowledge of the required operating environment	Evaluation during the product installation
		M 1a.2.11	Efficacy of the guide	Evaluation during the product installation
Q 1a.3	*What degree of replaceability does the product offer?*	–	*To be defined during the instantiation*	*To be defined during the instantiation*
Q 1a.4	*What degree of coexistence does the product offer?*	–	*To be defined during the instantiation*	*To be defined during the instantiation*

Table 5
Questions and metrics in the EFFORT measurement framework pertaining Sub-goal 1d.

SUB-GOAL 1d - Functionality				
Question		Metric		
Id	Name	Id	Name	Application and data source
Q 1d.1	What degree of functional adequacy does the product offer?	–	*To be defined during the instantiation*	*To be defined during the instantiation*
Q 1d.2	What degree of interoperability does the product offer?	M 1d.2.1	Level of data importability	Evaluation of the number of standard formats available for the data import
		M 1d.2.2	Level of data exportability	Evaluation of the number of standard formats available for the data export
Q 1d.3	What degree of functional accuracy does the product offer?	–	*To be defined during the instantiation*	*To be defined during the instantiation*

Table 6
Questions in the EFFORT measurement framework pertaining to community trustworthiness.

Goal 2 – Community trustworthiness				
Question		Metric		
Id	Name	Id	Name	Application and data source
Q 2.1	How many developers does the community involve?	M 2.1.1	Number of committers	Analysis of the official repository and query of the ohloh web site
Q 2.2	What degree of activity does the community have?	M 2.2.1	Number of major releases per year	Analysis of the official project web site
		M 2.2.2	Average number of commits per year	Analysis of the official repository and query of the ohloh web site
		M 2.2.3	Average number of commits per committer	Analysis of the official repository and query of the ohloh web site
		M 2.2.4	Closed bugs index	Analysis of the official project tracker
		M 2.2.5	Index of satisfied requests	Analysis of the official project tracker
Q 2.3	Are the support tools available and effective?	M 2.3.1	Average number of threads per year	Analysis of the official project forum
		M 2.3.2	Index of unanswered threads	Analysis of the official project forum
		M 2.3.3	Number of forums	Analysis of the official project forum
		M 2.3.4	Average number of threads per forum	Analysis of the official project forum
		M 2.3.5	Average number of posts per year	Analysis of the official project forum
		M 2.3.6	Forum internationalisation level	Analysis of the official project forum
		M 2.3.7	Number of trackers	Analysis of the official project tracker
		M 2.3.8	Volume of wikis	Analysis of the official project wiki
		M 2.3.9	Number of faqs	Analysis of the official project web site
Q 2.4	Are support services provided?	M 2.4.1	Availability of training services	Analysis of the official project web site
		M 2.4.2	Temporal coverage of training services	Analysis of the official project web site
		M 2.4.3	Availability of e-learning services	Analysis of the official project web site
		M 2.4.4	Availability of phone assistance	Analysis of the official project web site
		M 2.4.5	Availability of certification services	Analysis of the official project web site
		M 2.4.6	Availability of outsourcing services	Analysis of the official project web site
		M 2.4.7	Availability of maintenance services	Analysis of the official project web site
		M 2.4.8	Availability of information and services for TCO estimation	Analysis of the official project web site
		M 2.4.9	Availability of consulting services	Analysis of the official project web site
Q 2.5	Is the documentation exhaustive and easily consultable?	M 2.5.1	Number of topics covered in the administrator documentation	Analysis of the official project wiki
		M 2.5.2	Number of topics covered in the user documentation	Analysis of the official project documentation
		M 2.5.3	Number of topics covered in the technical documentation	Analysis of the official project documentation
		M 2.5.4	Number of topics covered in the other documents	Analysis of the official project wiki
		M 2.5.5	Number of additional documentation files	Analysis of the official repository and project wiki
		M 2.5.6	Usability of the documentation	Analysis of the official project documentation

each project, the source code was analysed and the product itself was used. Additional data sources were some useful web sites, such as *sourceforge.net, freshmeat.net* and *ohloh.net*. The "in vitro" nature of the analysis did not permit a realistic evaluation of the efficiency of GOAL1, so it has not been considered here.

The sources used for gathering data have different natures for the three goals. In particular, they are specifically related to the analysed software system components, such as documentation, source code, and installation process, as shown in Tables 4 and 5. Table 6 shows that the evaluation of the community requires the analysis of sources related to the specific software system project, such as its forum, mailing lists, bug tracker, and project wikis. Finally, the third goal can be partially evaluated by considering the official open source project site, such as *sourceforge.net*, *freshmeat.net*, and *code.google.com*. The last subsection discusses how the metrics can be aggregated to answer the questions quantitatively.

Table 7
Questions in the EFFORT measurement framework pertaining to product attractiveness.

Goal 3 – Product attractiveness				
Question		Metric		
Id	Name	Id	Name	Application and data source
Q 3.1	What degree of functional adequacy does the product offer?	–	–	–
Q 3.2	What degree of diffusion does the product achieve?	M 3.2.1	Number of downloads	Data available from sourceforge project section
		M 3.2.2	Freshmeat popularity index	Data available from freshmeat project section
		M 3.2.3	Number of rating source forge users	Data available from sourceforge project section
		M 3.2.4	Positive rating index	Data available from sourceforge project section
		M 3.2.5	Number of success stories	Analysis of the success story on the official web site
		M 3.2.6	Google visibility	Available from the by query on google google.com (software name + project category)
		M 3.2.7	Number of official partners	Analysis of the official web site
		M 3.2.8	Number of published books	Available from the by query on amazon.com (software name + project category)
		M 3.2.9	Number of citations by domain expert	Identification of the expert and article analysis
		M 3.2.10	Number of academic publications	Available from the query on google scholar (software name + project category)
		M 3.2.11	Sponsor availability	Analysis of the official project web site
Q 3.3	What level of cost-effectiveness is estimated?	M 3.3.1	Availability of services and information for the estimation of the TCO	Analysis of the official project web site
		M 3.3.2	Availability of an edition without license cost	Analysis of the official project web site
		M 3.3.3	Cost of the minimal edition	Analysis of the official project web site
		M 3.3.4	Cost of the complete edition	Analysis of the official project web site
Q 3.4	What degree of reusability and redistribution is left by the license?	M 3.4.1	License type	Analysis of the official project web site

3.1. Product quality

Product quality is one of the main aspects that dictate the quality of a project. Thus, *Goal 1* is defined as follows:

> *Analyse the **software product** with the aim of **evaluating** its **quality** from the **software engineer**'s point of view.*

With respect to the ISO/IEC 9126 standard, product quality is defined on the basis of the following characteristics:

- **Portability** – The capability of the software product to be transferred from one environment to another.
- **Maintainability** – The capability of the software product to be modified. Modifications may include corrections, improvements or adaptation of the software to changes in environment and in requirements and functional specifications.
- **Reliability** – The capability of the software product to maintain a specified level of performance when used under specified conditions.
- **Functionality** – The capability of the software product to provide functions that meet stated and implied needs when the software is used under specified conditions.
- **Usability** – The capability of the software product to be understood, learned, used and attractive to the adopter when used under specified conditions.
- **Efficiency** – The capability of the software product to perform appropriately, relative to the amount of resources used, under stated conditions.

Table 3 includes the sub-goals defined for each characteristic of Goal 1 except in-use quality. The table also lists all the questions defined for each sub-goal. Tables 4 and 5 list the metrics that are used for answering the questions of sub-goals 1.a and 1.d regarding portability and functionality. The choice of the metrics was performed by considering parts 2 and 3 of the ISO/IEC Standard 9126 [31]. In addition, Tables 4 and 5 indicate which data source can be considered to evaluate each metric. This added information makes the evaluation framework an operative tool that is applicable in a working context.

Nevertheless, as Tables 4 and 5 show, Goal 1 cannot be completely defined without knowing some characteristics and the application domain of the software systems to be evaluated. The questions that need to be instantiated to a specific context are indicated in Table 2 in *italics*. For example, with respect to portability, some sub-characteristics, such as replaceability and adaptability, are more strategic for some application domains than for other. Therefore, the identification of the metrics needed to perform this evaluation implies knowledge of the specific context of use of the software system to be evaluated. Additional instantiation activities have to be performed to evaluate the functionality, considering the users' stated and implied needs, which depend on the application domain. Moreover, evaluating functionality requires the evaluation of some functional aspects, such as adequacy and accuracy, that are dependent on the application domain of the analysed software system. Similarly, the evaluation of maintainability implies a dependence of the measurement framework on the programming paradigm used that can guide the choice of the specific metrics to be considered. For example, the types of software systems analysed in this paper required the use of metrics related to an object-oriented paradigm [13]. This does not represent a restriction, even if the full definition of the metrics may require an instantiation of the measurement framework to evaluate that specific type of software system. For these reasons, only a few general-purpose metrics are included in the baseline version of EFFORT. Obviously, additional measures can be performed alongside the indicated ones.

3.2. Community trustworthiness

Community trustworthiness indicates the degree of trust that a user has in a community with respect to the support offered. Communities can provide support by means of good execution of the development activity; use of tools, such as wikis, forums, and trackers; and provision of services, such as maintenance, certification, consulting and outsourcing, and documentation. *Goal 2* is defined as follows:

> *Analyse the **support** offered with the aim of **evaluating** the **community** with respect to the **trustworthiness** from the **(user or organisation) adopter**'s point of view.*

As already explained, the motivation for this goal comes from the fact that the main concerns of an OSS potential adopter pertain to the support and services he can receive when installing and using the system and the availability of new versions of the system. This aspect was also confirmed in interviews conducted with users and experts of Open Source Software systems. The purpose of the goal is to evaluate whether a community offers an adequate level of support.

Questions and metrics pertaining to *Goal 2* are shown in Table 6. The table also includes the source and modality of the evaluation of each metric. Questions and metrics pertaining to Goal 2 were identified by analysing other open source quality models, specifically, projects QSOS [12], OpenBRR [41] and QualiPSo [16,17] that focus on community quality.

3.3. Product attractiveness

The third goal of EFFORT is to evaluate the attractiveness of the product from the perspective of the user community. The term *attractiveness* refers to all the factors that influence the adoption of a product by a potential user, who perceives the convenience and usefulness of the product in achieving his goals.

Goal 3 pertains to product attractiveness and is formalised as follows:

> *Analyse the **software product** with the aim of **evaluating** it in terms of its **attractiveness** from the **(user or organisation) adopter**'s point of view.*

Two elements that must be considered in the selection of a F*l*OSS product are *functional adequacy* and *diffusion*. The latter could be considered an indicator of how the product is appreciated and recognised as useful and effective. Other factors to be considered are *cost-effectiveness*, the Total Cost of Ownership (TCO) [34], and the *type of license*.

Questions and metrics pertaining to Goal 3 are shown in Table 7. They were formulated by analysing the other proposed quality models, with particular reference to the studies developed in the QualiPSo project [17]. The results presented in [38,45] were also considered for use in analysing the relations between attractiveness and source code metrics.

Goal 3 is the goal that is most dependent on the application context, because every type of software product is developed to satisfy different needs. For instance, the more efficient a real-time software program is, the more attractive it is, or with respect to an ERP system, the more configurable and customisable the system is, the more attractive it is. Such things are not necessarily true for a word processing program, where the user requires ease of use and compliance to de facto standards. Therefore, Goal 3 depends greatly on the application domain of the analysed software system and must be customised to the specific context.

The customisation of question Q3.1 has to be addressed in the same way as for Goal 1. Additional customisation involves the addition of new specific context questions. The fifth column of Table 7 indicates sources and modalities for evaluating the metrics.

3.4. Data analysis

The collected data are aggregated according to the interpretation given to them, so that one can obtain useful information for answering the questions. Moreover, aggregation for each question gives an indication of the achievement of the goals. In conducting aggregation, the following issues need to be considered:

- Metrics have different types of scales, depending on their nature. Thus, it is not always possible to directly aggregate measures. To overcome this, after the measurement is performed, each metric is mapped to a discrete score in the [1–5] range, where 1 = inadequate, 2 = poor, 3 = sufficient, 4 = good, and 5 = excellent.
- A high value of a metric can be interpreted in a positive or negative way, according to the context of the related question; the same metric could even contribute in two opposite ways, depending on the context of two different questions. So, the appropriate interpretation needs to be provided for each metric.
- Depending on the evaluator's point of view, questions do not all have the same relevance in the evaluation of a goal. Therefore, a relevance marker can be associated with each metric in the form of a numeric value in the [1–5] range. A value of 1 is associated with questions of minimum relevance, while a value of 5 is associated with questions of maximum relevance.

The aggregation function for Goal g is defined as follows:

$$q(g) = \frac{\sum_{q \in Q_q} r_q * m(q)}{\sum_{q \in Q_q} r_q} \tag{1}$$

where r_q is the relevance associated with question q (sub-goal for Goal 1), Q_g is the set of questions (sub-goals for Goal 1) related to Goal g, and $m(q)$ is the aggregation function of the metrics of question q. In particular:

$$m(q) = \frac{\left\{\sum_{id \in M_q} i(id) * v(id) + [1 - i(id)] * [v(id) \bmod 6]\right\}}{|M_q|} \tag{2}$$

where $v(id)$ is the score obtained for metric id and $i(id)$ is used to give a positive or negative interpretation to the metric with respect to the q question. In particular:

$$i(id) = \begin{cases} 0 & \text{if the metric has a negative interpretation} \\ 1 & \text{if the metric has a positive interpretation} \end{cases} \tag{3}$$

M_q is the set of metrics related to question q.

As shown above, EFFORT includes the definition of the relevance, r_{id}, associated with each question id in the evaluation of the goals. The relevance markers refer exclusively to the Open Source Software and do not consider the application context. Obviously, depending on the evaluator's point of view, the relevance attributed to a question can vary. A further instantiation task involves the definition of the question relevance values.

3.5. EFFORT customisation

To achieve a more focused evaluation, the proposed EFFORT framework can be customised, taking into account a careful characterisation of the open source software application domain. For example, it has been customised to the Customer Relationship Management (CRM) domain for analysing F*l*OSS CRM systems [3].

Customisation requires the execution of the following steps:

Analysis of the application domain – additional information pertaining to the specific context of the open source software considered is acquired in this step. With this in mind, interviews with the user/adopter must be conducted to better understand the specific interests, expectations and doubts arising from the adoption of a specific Open Source Software system.

Validation of all existing questions – the information collected in the previous step is validated in this step to determine whether it can be supported by the EFFORT baseline version. The domains of the possible values adopted for each metric are also checked to validate their soundness in the specific context of interest.

Introduction of new questions – it could emerge from the previous steps that some questions are not considered in the baseline framework, or if they are included, the question set does not consider all the expected specific aspects. On the basis of this analysis, new specific questions need to be introduced.

As a consequence of this customisation process, the framework instantiation can be performed at the level of questions or goals. In particular, it is possible to execute an *integration* task by better specifying the existing questions in terms of further metrics that can be discovered using the additional knowledge of the stated application context, while the intervention at the goal level, the *extension* of goals, is accomplished by adding more questions.

During the instantiation of EFFORT, a relevance strictly referring the application domain and point of view can be associated with each metric by using a numeric value in the [1–5] range. The definition of these new relevance markers depends on the experience and knowledge of the software engineer with respect to the specific domain of the software system to be analysed. Therefore, the new markers are combined with those defined in EFFORT. A new aggregation function for Goal g is defined as follows:

$$q(g) = \frac{[\sum_{q \in Qg}(rFlOSS_q + rSC_q) * m(q)]}{\sum_{q \in Qg}(rFlOSS_q + rSC_q)} \quad (4)$$

where $rFlOSS_q$ represents the relevance marker in the FlOSS context associated with question q, or sub-goal q of Goal 1; rSC_q indicates the relevance indicator in the specific context associated with question q, or sub-goal q of Goal 1; Q_g is the set of questions (sub-goals

Table 8
Integration of the measurement framework during the EFFORT instantiation.

Framework integration				
Id question	Question	Id metric	Metric	Application and Data Source
Q 1a.1	What degree of adaptability does the product offer?	M 1a.1.2	Number of supported DBMS	Analysis of the project technical documentation
		M a1.1.3	Availability of a client WEB interface	Analysis of the official project web site
Q 1a.3	What degree of replaceability does the product offer?	M 1a.3.1	Availability of functionality for the creation of data backup	Evaluation by the product use and analysis of the documentation
		M 1a.3.2	Availability of functionality for the restoration of data backup	Evaluation by the product use and analysis of the documentation
		M 1a.3.3	Availability of backup services	Analysis of the data on the official web site
		M 1a.3.4	Number of file formats for the reporting	Evaluation by the product use and analysis of the documentation
Q 1b.1	What degree of analysability does the product offer?	M 1b.1.7	Javadoc density	Analysis of the javadoc provided
Q 1c.2	What degree of recoverability does the product offer?	M 1c.2.4	Availability of tools for the management of transactions	Software analysis and analysis of the documentation
Q 1d.1	What degree of functional adequacy does the product offer?	M 1d.1.1	Availability of a module for financial and accounting management	Evaluation by product use and analysis of the documentation
		M 1d.1.2	Availability of a module for document management	Evaluation by product use and analysis of the documentation
		M 1d.1.3	Availability of a portal for the management of customer–supplier management	Evaluation by product use and analysis of the documentation
		M 1d.1.4	Availability of a module for sales management	Evaluation by product use and analysis of the documentation
		M 1d.1.5	Availability of a module for warehouse management	Evaluation by product use and analysis of the documentation
		M 1d.1.6	Availability of a module for project management	Evaluation by product use and analysis of the documentation
		M 1d.1.7	Availability of a module for purchase management	Evaluation by product use and analysis of the documentation
		M 1d.1.8	Availability of a module for production management	Evaluation by product use and analysis of the documentation
		M 1d.1.9	Availability of a module for human resource management	Evaluation by product use and analysis of the documentation
Q 1d.2	What degree of interoperability does the product offer?	M 1d.2.3	Availability of web-services support	Analysis of the official documentation
Q 3.1	What degree of functional adequacy does the product offer?	–	Metrics of Question Q 1d.1	–
Q 3.3	What is the estimated degree of affordability?	M 3.3.5	Migration cost among different versions	Evaluation by product use and analysis of the documentation
		M 3.3.6	Data population cost of the system	Evaluation by product use and analysis of the documentation
		M 3.3.7	System configuration cost	Evaluation by product use and analysis of the documentation
		M 3.3.8	System customisation cost	Evaluation by product use and analysis of the documentation

of Goal 1) related to Goal g; and $m(q)$ is the aggregation function of the metrics of question q presented in the previous sub-section.

The definition of the relevance markers is needed to better fit the evaluation by considering the user/adopter needs and interests.

4. EFFORT application

This section demonstrates how EFFORT can be applied to evaluation of Open Source Software systems. The chosen application domain was open source ERP due to the growing interest of the related communities. This choice requires the customisation of EFFORT to the ERP application domain. The customised EFFORT framework represents a complete model for evaluating the quality of F*l*OSS ERP systems with respect to both product and community. In particular, specific questions and metrics concerning the ERP context were added to the baseline version of EFFORT.

The customisation of EFFORT was followed by the evaluation of five F*l*OSS ERPs, and the results are described below.

With respect to the selection of the projects to be evaluated, the most diffused ones among ERP F*l*OSS systems developed using Java technology were identified. Java technologies were considered for two reasons. First, for the purpose of obtaining a realistic comparison of the characteristics involving static analysis of code, it was easier to make comparisons of products that were implemented according to the same paradigm, or even better, using the same technology. This does not mean that it is not possible to compare heterogeneous projects, but there are some limitations in doing so. For example, different programming languages have different expressional power and different amounts of code are necessary to implement the same functionality in different languages. The second reason pertains to the availability of static analysis tools, which is considerable for Java technology. The projects selected were *Compiere* (www.compiere.com), *Adempiere* (www.adempiere.org), *Openbravo* (www.openbravo.com), *OFBiz* (ofbiz.apache.org) and *JFire* (www.jfire.org). The next subsection describes the customisation of EFFORT, and the subsequent section discusses the results achieved.

4.1. Customising EFFORT to the ERP context

The comparison of the existing ERP selection studies reported in Table 2 was particularly useful during the ERP customisation process. Indeed, the criteria listed in Table 2 synthesise the point of view of experts in the ERP domain. Thus, in accordance with the customisation steps described in the previous section, specific metrics and/or questions were introduced for the criteria listed in the table and not considered in the EFFORT baseline version. In the customisation process, *integration* and *extension* tasks were executed. *Integration* pertains to adding additional metrics for answering some EFFORT baseline questions, while *extension* was applied to some goals by adding more questions.

In particular, with respect to the integration, Table 8 shows all the metrics that were added for answering some baseline questions. The greater part of the integration concerned Goal 1: product quality. Particular attention was paid to the strategic role of data and the necessity of integrating an ERP application with the exist-

Table 9
Extension of Goal 3 of the EFFORT measurement framework for evaluating ERP systems.

Extension of Goal 3				
Id question	Question	Id metric	Metric	Application and Data Source
Q 3.5	What degree of support for migration between different releases is offered?	M.3.5.1	Availability of functionality for creation of data backup	Evaluation by product use and analysis of the documentation
		M.3.5.2	Availability of functionality for restoration of data backup	Evaluation by product use and analysis of the documentation
		M.3.5.3	Availability of backup services	Analysis of the data on the official web site
		M.3.5.4	Availability of documentation of migration between versions	Analysis of the data on the official web site
		M.3.5.5	Availability of automatic migration tools	Evaluation by product use and analysis of the documentation
		M.3.5.6	Availability of documentation for migrating the database	Analysis of the data on the official web site
Q 3.6	What degree of support for population of the system is offered?	M.3.5.1	Number of standard formats for importing data	Evaluation by product use and analysis of the documentation
Q 3.7	What degree of support for configuration of the system is offered?	M.3.7.1	Availability of a wizard for configuring the system	Evaluation by product use and analysis of the documentation
		M.3.7.2	Number of charts of accounts	Analysis of the data on the official web site
		M.3.7.3	Number of supported languages	Analysis of the data on the official web site
		M.3.7.4	Availability of functionality for tax category management	Evaluation by product use and analysis of the documentation
		M.3.7.5	Multicurrency coverage index	Evaluation by product use and analysis of the documentation
		M.3.7.6	Availability of documentation for supporting the starting setup	Analysis of the data on the official web site
		M.3.7.7	Availability of documentation for supporting language configuration	Analysis of the data on the official web site
		M.3.7.8	Availability of documentation for supporting tax category configuration	Analysis of the data on the official web site
		M.3.7.9	Availability of documentation for supporting the configuration of the chart of accounts	Analysis of the data on the official web site
Q 3.8	What degree of support for customisation of the system is offered?	M.3.8.1	Availability of functionality for installing an extension of the user interface	Evaluation by product use and analysis of the documentation
		M.3.8.2	Availability of functionality for creating a new module of the user interface	Evaluation by product use and analysis of the documentation
		M.3.8.3	Availability of functionality for customising the user interface	Evaluation by product use and analysis of the documentation

ing information systems of an organisation. In particular, considering the criteria listed in Table 2, it emerged that the product quality for ERP systems should focus specifically on *functionality*, *usability*, *system reliability*, *interoperability*, *integration/modularity*, *implementation time*, *customisation and flexibility*, *migration*, and *technical quality*. Tables 8 and 9 show the additional metrics that were considered for addressing these aspects in the framework. Specifically, Table 8 shows that the framework sections regarded adaptability and replaceability (and, consequentially, portability) were extended. In particular, the number of supported Data Base Management Systems (DBMS) and availability of a web client interface were considered for the adaptability characteristic, whereas availability of functionality for backup and restoration of data, availability of backup services and number of reporting formats were taken into account for the replaceability characteristic. Evaluating the analysability required the adding of a Javadoc density metric, because all the analysed ERP systems were based on the Java technology. In addition, because the analysed software systems were based on an object-oriented paradigm, metrics related to this paradigm were considered. This required the instantiation of the measurement framework with the adoption of specific object-oriented metrics. A metric was introduced for evaluating *recoverability. This metric considered the availability of tools for transaction management* (M1.c.2.4), which is an issue of concern with ERP systems. A set of metrics, from M1d.1.1 to M1d.1.9, were added for evaluating functional adequacy. The metrics concerned the partial or total existence of specified modules. Further customisation could be performed for analysing the functional adequacy of individual modules in the execution of specific operations.

With respect to Goal 3, Table 8 presents the integration performed for understanding the economic advantage when an ERP F*l*OSS system is adopted. The integration required the addition of metrics concerning costs in the questions of Goal 3. In fact, the EFFORT baseline just considered the possibility of obtaining the product free of charge and having to pay an amount for an annual subscription. Because this is not sufficient for ERP systems, a customisation was considered to include costs for migration between releases (M 3.3.5), population of the system (M 3.3.6), customisation (M 3.3.8), and configuration (M 3.3.7).

In addition, the criteria listed in Table 2 were further considered to drive an additional customisation of Goal 3. Specifically the criteria affecting Goal 3 are *functionality*, *usability*, *support services*, *system reliability*, *interoperability*, *references and reputation of vendors*, *partnership*, *integration/modularity*, *implementation time*, *customisation and flexibility*, and *migration*. Goal 3 is the most dependent on the application domain and regards the way a software system should be used for being attractive. This aspect does not depend only on the license costs but also on the costs of both adapting the software system to specific user needs and maintaining it by installing or updating versions or adopting new releases. Therefore, including the criteria above in the evaluation also required an extension intervention.

Goal 3 was extended by inserting the additional questions listed in Table 9. In particular, the following aspects were considered:

- *Migration between different versions of the software*, in terms of support provided for switching from one release to another one. In the context of ERP systems, this migration cannot be accomplished like a new installation because it would be too costly, considering that such a system is generally customised and hosts a large quantity of data.
- *System population*, in terms of support offered for importing large volumes of data.
- *System configuration*, which refers to the support provided in terms of functionality and documentation pertaining to the adaptation of the system to user needs, such as localisation and internationalisation. The greater this support is, the shorter the start-up time is.
- *System customisation*, which pertains to support provided without direct access to the source code to make changes to the system, such as the definition of new modules, installation of extensions, personalisation of reports and creation of new workflows. This characteristic is very desirable in ERP systems.

Table 9 shows the questions extending Goal 3 and their related metrics. Each new question is related to one of the listed characteristics. Table 9 also lists the data source of each metric.

4.2. Results

This section presents the results obtained from applying both the baseline and customised version of EFFORT to the following five ERP F*l*OSS systems: Compiere, Adempiere, Openbravo, OFBiz and JFire. Fig. 3 summarises the comparison among the product qualities of the selected projects. The results indicated that the software product of the Compiere project is the best, followed by Adempiere. The project that received the worse score was JFire. The graph shows that the personalisation of EFFORT does not influence the trends of the comparison, even though the results were obviously different. All projects except OFBiz yielded a lower score when the personalised version of EFFORT was considered.

Fig. 4 shows the results obtained using the customised version of EFFORT for the main sub-characteristics of product quality. Additional information is provided in Table 10, where ECV stands

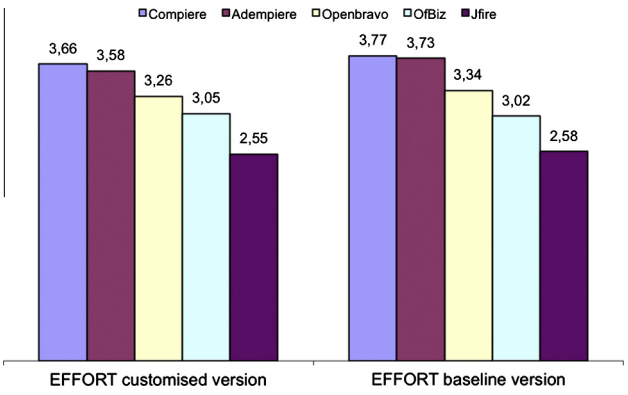

Fig. 3. Comparison of the product quality of the five evaluated OSS ERP systems.

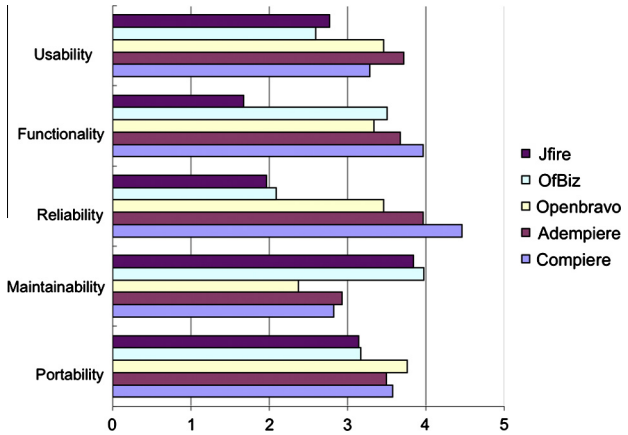

Fig. 4. Detailed results pertaining to the product quality of the five evaluated OSS ERP systems.

Table 10
Results of the evaluation of the product quality of the five analysed OSS ERP systems.

Quality characteristic	Relevance		Score									
	FlOSS	ERP	Compiere		Adempiere		Openbravo		OFBiz		JFire	
			EBV	ECV	EBV	ECV	EBV	ECV	EBV	ECV	EBV	ECV
Portability	3	2	4.10	3.57	3.83	3.49	4.45	3.76	3.41	3.16	3.36	3.14
Adaptability			5.00	3.33	5.00	3.67	5.00	3.67	5.00	4.33	5.00	4.33
Installability			2.64	2.64	2.82	2.82	3.36	3.36	3.91	3.91	4.09	4.09
Replaceability			4.67	4.75	3.67	4.00	5.00	4.25	1.33	1.25	1.00	1.00
Maintainability	3	4	2.83	2.83	2.93	2.93	2.37	2.37	3.97	3.97	3.84	3.84
Analyzability			3.00	3.00	2.43	2.43	1.57	1.57	3.71	3.71	3.57	3.57
Changeability			2.80	2.80	3.40	3.40	2.40	2.40	4.40	4.40	4.40	4.40
Testability			2.50	2.50	2.88	2.88	1.50	1.50	3.75	3.75	4.38	4.38
Technology cohesion			3.00	3.00	3.00	3.00	4.00	4.00	4.00	4.00	3.00	3.00
Reliability	3	5	4.42	4.46	4.17	3.96	3.58	3.46	2.00	2.08	1.83	1.96
Robustness/Maturity			4.17	4.17	4.67	4.67	2.17	2.17	2.67	2.67	2.67	2.67
Recoverability			4.67	4.75	3.67	3.96	5.00	4.75	1.33	1.50	1.00	1.25
Functionality	5	5	4.13	3.96	3.92	3.67	3.00	3.33	3.17	3.50	1.67	1.67
Functional adequacy			3.25	3.25	3.33	3.33	3.00	3.00	3.33	3.33	2.33	2.33
Interoperability			5.00	4.67	4.50	4.00	3.00	3.67	3.00	3.67	1.00	1.00
Usability	4	4	3.28	3.28	3.72	3.72	3.46	3.46	2.59	2.59	2.76	2.76
Pleasantness			2.00	2.00	3.00	3.00	4.00	4.00	2.00	2.00	4.00	4.00
Operability			4.00	4.00	4.00	4.00	3.75	3.75	3.25	3.25	3.50	3.50
Understandability			3.89	3.89	4.11	4.11	3.33	3.33	2.88	2.88	2.56	2.56
Learnability			3.25	3.25	3.75	3.75	2.75	2.75	2.25	2.25	1.00	1.00
Product quality	*(EBV)*		*3.77*		*3.73*		*3.34*		*3.02*		*2.58*	
	(ECV)			*3.66*		*3.58*		*3.26*		*3.05*		*2.55*

for EFFORT Customised Version and EBV stands for EFFORT Baseline Version.

Compiere had the best values for functionality and reliability and also a very good degree of usability and portability. Its maintainability results are less than adequate, as were those for Adempiere and Openbravo. The products that offer less global quality provide better value in maintainability, mostly because they are smaller than the others. Despite its size, OFBiz had a good functionality value, while JFire did not. The latter also had the worst score in portability, even though it is the easiest system to install. This is because JFire has very poor replaceability, in terms of backup/recovery functionalities and services and the number of import/export standard formats for reports. Openbravo was found to be the most portable product, while Adempiere had the best value in usability, operability and understandability, mainly because it offers at almost every point the possibility of undo/redo operations, visibility of the system and operation state, good coherence in presentation and a graphical environment across operations, clarity in texts and titles, a significant presence in online help and good user documentation.

Fig. 5 shows that Compiere Community obtained one of the worst community trustworthiness scores, just below JFire's score.

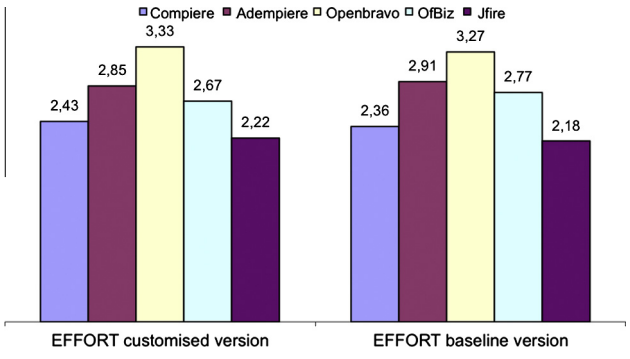

Fig. 5. Comparison of the community trustworthiness of the five evaluated OSS ERP systems.

This is due to its poor documentation availability and insufficient community activity, particularly in development activities. In fact, the average commits per year and total number of commits are low compared to those of the other systems, even though there is quite a good percentage of bugs fixed on the total opened ones. Looking at Fig. 5 and Table 11, one can observe that Openbravo appears to be supported by the most trusted community. In fact, it offers a good amount of easily consultable documentation especially for system administrators.

There are many developers working on the Openbravo project, and support tools are largely used (only Adempiere gave better results for that). In particular, there are many forum posts per year and forums are available. The Wiki is quite large, and several FAQs are provided. Openbravo also offers comprehensive support by means of consulting, training and outsourcing services. Activity in development is not as intensive, taking into account the community size. A barely sufficient trust score can be given to the Adempiere community, even though its results place it second in the comparison. JFire was rated lowest in community trustworthiness. OFBiz almost everywhere presents medium scores. A positive note for OFBiz comes from the community activity, for which it has the best value. As Fig. 5 shows, the trends of the comparison are not influenced by the use of the customised version of EFFORT instead of the baseline one, as was found for product quality.

With respect to product attractiveness, Fig. 6 shows that Openbravo obtained the best score, followed by Compiere. For the EFFORT baseline version, the opposite was true. In this case, different trends in the comparison were obtained using the different versions of the framework, as can also be observed by comparing Adempiere to OFBiz. Both OFBiz and JFire had worse results when the EFFORT customised version was used, unlike what happened with the other products. Detailed results pertaining to product attractiveness are reported in Table 12. The most customisable and configurable product is Openbravo, which has, on the other hand, a low value of functional adequacy. Only JFire has a lower score of functional adequacy, and it generally was the worst in other respects. The most diffused product is definitely Compiere, followed by Openbravo. These two products also have the best values of legal reusability, because they offer the possibility of choos-

Table 11
Results of the evaluation of the community trustworthiness of the five analysed OSS ERP systems.

Quality characteristic	Relevance		Score				
	FlOSS	ERP	Compiere	Adempiere	Openbravo	OFBiz	JFire
Developers number	2	1	2.00	2.00	4.00	2.00	2.00
Community Activity	4	2	2.60	2.60	2.80	3.75	2.50
Support tools	5	4	2.44	3.89	3.22	3.00	1.78
Support services	2	4	3.44	1.78	4.11	1.89	2.78
Documentation	4	4	1.67	3.00	3.00	2.33	2.17
Community trustworthiness	*(EBV)*		*2.36*	*2.91*	*3.27*	*2.77*	*2.18*
	(ECV)		*2.43*	*2.85*	*3.33*	*2.67*	*2.22*

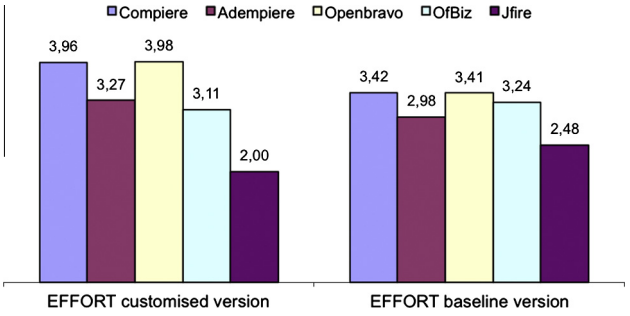

Fig. 6. Comparison of the product attractiveness of the five evaluated OSS ERP systems.

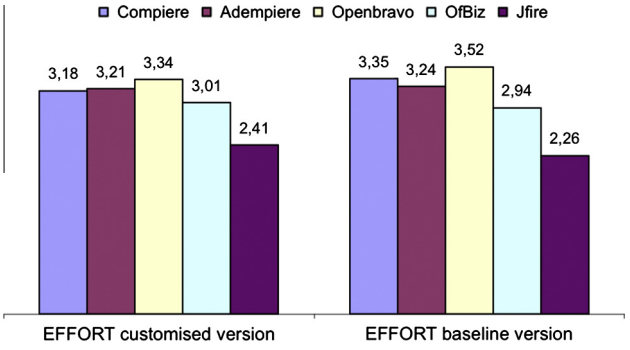

Fig. 7. Comparison of the project global quality of the five evaluated OSS ERP system.

ing different types of license, even a commercial one. The products with the best functional adequacy are Adempiere and OFBiz.

Fig. 7 includes a comparison among the aggregated results that represent the global quality of the projects. According to both EFFORT versions, the best FlOSS ERP project is Openbravo. Compiere and Adempiere have similar results, even if the former appears a little better in the customised version of the framework. Table 13 shows the cumulative results for each analysed product and evaluated characteristic. The table shows that the results obtained for the systems using the EFFORT Baseline Version (EBV) are comparable to those obtained using the EFFORT Customised Version (ECV), except for product attractiveness, for which higher values were obtained using ECV.

The interpretation of the results is largely subjective, because the point of view of the evaluator must be considered. Nevertheless, through the objectivity of the majority of the metrics, the specialisation of the EFFORT fits the framework to the users' needs. Different professional evaluators may obtain different results, but this depends on their expectations as expressed through the relevance markers.

4.3. Industrial discussions

To obtain feedback on the EFFORT model from the potential adopters of OSS ERP, a preliminary survey was conducted. Specifically, a workshop was organised to disseminate the model itself and interview potential adopters. A brief questionnaire (see Appendix A) was submitted to the workshop participants. Sixteen respondents successfully completed the questionnaire.

The survey considered two objectives. Other objectives could be considered, but a complete industrial validation is beyond the scope of this study. Thus, the primary objective was to determine whether the EFFORT model was considered effective for an open source selection process and how the respondents considered the EFFORT results obtained for selecting ERP systems. The second objective was to determine whether the respondents would consider the adoption of the EFFORT model when considering open source selection within their enterprises. Based on these two objectives, 8 questions were designed.

Table 12
Results of the evaluation of product attractiveness of the five analysed OSS ERP systems.

Quality characteristic	Relevance		Score									
	FlOSS	ERP	Compiere		Adempiere		Openbravo		OFBiz		JFire	
			EBV	ECV	EBV	ECV	EBV	ECV	EBV	ECV	EBV	ECV
Functional adequacy	5	5	3.25	3.25	3.33	3.33	3.00	3.00	3.33	3.33	2.33	2.33
Diffusion	4	3	4.00	4.00	2.27	2.27	3.45	3.45	2.36	2.36	1.64	1.64
Cost effectiveness	3	5	2.40	3.22	4.00	3.88	3.50	4.00	4.00	3.38	4.00	2.88
Legal reusability	1	5	5.00	5.00	1.00	1.00	5.00	5.00	4.00	4.00	2.00	2.00
Migrability	–	5	–	3.67	–	4.00	–	4.50	–	1.17	–	1.00
Data importability	–	5	–	5	–	5.00	–	4.00	–	5.00	–	1.00
Configurability	–	2	–	3.89	–	3.33	–	4.11	–	3.22	–	1.67
Customizability	–	4	–	4.67	–	4.00	–	5.00	–	2.00	–	1.67
Product attractiveness	*(EBV)*		*3.42*		*2.98*		*3.41*		*3.24*		*2.48*	
	(ECV)			*3.96*		*3.27*		*3.98*		*3.11*		*2.00*

Table 13
Aggregated results of the evaluation of the five analysed OSS ERP systems.

Quality characteristics	EFFORT version	Compiere	Adempiere	OpenBravo	OFBiz	JFire
Product quality	(EBV)	3.77	3.73	3.34	3.02	2.58
	(ECV)	3.66	3.58	3.26	3.05	2.55
Community trustworthiness	(EBV)	2.36	2.91	3.27	2.77	2.18
	(ECV)	2.43	2.85	3.33	2.67	2.22
Product attractiveness	(EBV)	3.42	2.98	3.41	3.24	2.48
	(ECV)	3.96	3.27	3.98	3.11	2.00

The survey targeted software engineers and enterprise managers involved with technology selection. The questionnaires, including both closed and open questions, were submitted by conducting interviews at the end of the dissemination workshop. Confidentiality and privacy were assured to all individuals returning the questionnaire and the organisations that they represented.

More than two thirds (67.7%) of the 16 organisations responded to our survey. They belonged to the local private commercial sector. The 16 organisations ranged from medium in size (18% of organisations) to very small sizes of less than 10 employees (82%).

Despite the small sample population, the consistency of the data obtained has increased our confidence in the applicability of the approach. Only 3 of the 16 organisations did not consider the EFFORT application satisfying, while 13 considered EFFORT an effective model that could be useful in supporting their OSS selection process. Most of these 13 organisations considered the results of ERP comparison consistent with their point of view. Finally, the majority of the respondents (70.8%) stated that they would consider adopting EFFORT adoption in their future OSS selection tasks.

Although the size of the survey sample was small, the authors believe that these 16 organisations provided enough sample data to support the planning of a broad survey of the applicability of EFFORT.

5. Conclusions

This paper proposes a quality model for the evaluation of Open Source Software systems and an operational framework to support the model.

The quality model consists of an evaluation framework and the guidelines for its application. The model considers both product quality and project quality in evaluating an OSS system. A number of characteristics were considered in relation to OSS products, such as the community driving the open source project and the openness of information that is useful for a more comprehensive evaluation of this type of software. The EFFORT framework was developed by considering these characteristics. In addition, EFFORT was developed with the following objectives in mind: covering the intersection among the other analysed models and overcoming their limitations by including other characteristics considered significant for Open Source Software. The results of this comparison can also be used to suggest future work in OSS evaluation.

The proposed framework is compliant with the ISO/IEC 9126 standard for product quality. It considers all of the characteristics defined by the ISO standard, except in-use quality. In addition, it considers the major aspects related to F*l*OSS projects.

Table 1 shows that EFFORT offers more coverage of the ISO standard than the other models proposed in the literature. In addition, EFFORT provides an operational framework for evaluating the quality product of a F*l*OSS system.

Fig. 1 shows a comparison between EFFORT and the other models analysed in this study. The characteristics shown in the figure concern the general aspects considered by the various OSS quality models, even though they formalise the characteristics in different ways.

Table 1 and Fig. 1 show that EFFORT is as complete as other OSS quality models, such as IRCA. In addition, the definition of EFFORT includes a proposed measurement framework that can be used to evaluate and compare alternative OSS solutions.

This paper also discusses the customisation of EFFORT to the evaluation of OSS software systems for specific application domains. Information about the working context of the software systems to be analysed can suggest the evaluation of additional aspects not included in the EFFORT baseline framework.

The framework is parametric with respect to the relevance that an evaluator attributes to the metrics evaluated. The quality model and measurement framework can accommodate different types of users: a potential adopter of a F*l*OSS system, such as an enterprise, that is mainly interested in the functionality it offers and its ease to use; a developer that wants to enhance it, who mainly considers aspects regarding the product quality and FlOSS project community providing support to the potential users and developers; or a potential sponsor who will be interested in the product's attractiveness, diffusion and functionality. Each potential user can assign relevance markers to the metrics on the basis of his/her evaluation interests. Thus, for example, a developer can assign higher values to the relevance markers of development and software engineering metrics, while a potential adopter can increase the influence of metrics regarding functionality, comprehension, and support offered.

The final result of the evaluation is a direct comparison of FlOSS systems with respect to the metrics emphasised (i.e., those with higher relevance markers), which facilitates the selection of the most suitable system for specific requirements.

To demonstrate the effectiveness and applicability of EFFORT, it was customised to fit the ERP software system domain.

The customisation process was based on an analysis of the models proposed in the literature for evaluating and selecting ERP software systems. The analysis of the criteria considered by these models permitted the definition of a framework that improves the quality models previously defined for selecting ERP systems. The comparison of the customised EFFORT with other models is shown in Table 2. The table shows that the customised EFFORT is an improvement over the Open Source ERP Guru model, which is the only model analysed that is focused on OSS ERP systems. Open Source ERP Guru is a useful platform for supporting the evaluation of ERP systems, even though it does not provide any operative tool for driving the selection. The customised EFFORT also includes characteristics considered by the other models that do not evaluate open source issues. As in Open Source ERP Guru, the only criteria included in Fig. 2 that are not considered by EFFORT are market-oriented criteria.

Customisation of the EFFORT framework required the introduction of additional metrics and better formalisation of other characteristics. Future work should consider the definition of mechanisms to extend and customise EFFORT to better characterise all of the aspects dependent on the application domain.

The applicability of the EFFORT framework and its customisation was investigated through the analysis of five F*l*OSS ERP systems. The results obtained led to the improvement of the measurement framework and identification of the F*l*OSS ERP system that offers the best quality with respect to a predefined set of relevance markers. The results showed that Compiere is the F*l*OSS ERP system that offers the best quality with respect to product quality and product attractiveness characteristics, while the community of OpenBravo provides the highest trustworthiness. The application of the EFFORT measurement framework to the evaluation and selection of an OSS system may significantly reduce the amount of analysis that must be conducted before adopting a system, with consequent savings in the time and cost required for gathering and interpreting data.

Obviously, the evaluation of the adequacy of an OSS system depends greatly on how the users use it. Therefore, the EFFORT framework also considers the adopter's point of view by defining the relevance markers associated with each metric and question in the data aggregation process.

Future investigation will consider the extension of the framework with questionnaires and other tools for evaluating customer satisfaction. This obviously needs to include a more complex analysis. In particular, methods and techniques customised for exploiting this aspect will be explored and defined. In addition, with reference to the ISO standard, further investigations will be performed with respect to the in-use quality characteristic. Furthermore, the authors will continue to search for additional evidence of the usefulness and applicability of EFFORT and its customisations by conducting additional studies also involving subjects working in specific operational contexts.

Appendix A. Survey questionnaire

(1) To what sector does your organisation belong?
Government
Public non-commercial organisation
Local private commercial organisation
Overseas-based private commercial organisation
Joint venture between public and private sectors

(2) How large is your organisation?

(3) Overall, how satisfied are you with the EFFORT approach?
Very dissatisfied
Dissatisfied
Somewhat satisfied
Very satisfied
Extremely satisfied

(4) Overall, how satisfied are you with EFFORT results?
Very dissatisfied
Dissatisfied
Somewhat satisfied
Very satisfied
Extremely satisfied

(5) Will you consider the use of EFFORT in your selection process?
Definitely will
Most likely will
Might or might not
Most likely will not
Definitely will not
Never used

(6) EFFORT can be satisfactory in addressing an ERP selection problem:
Strongly disagree
Somewhat disagree
Neutral
Somewhat agree
Strongly agree

(7) How likely are you to recommend EFFORT to others?
Definitely will recommend
Most likely will recommend
Might or might not recommend
Most likely will not recommend
Definitely will not recommend
Never used

(8) Please describe any particular aspect of the EFFORT that stood out:

References

[1] L. Aversano, I. Pennino, M. Tortorella, Evaluating the quality of FREE/OPEN source project, in: INSTICC Evaluation of Novel Approaches to Software Engineering conferences – ENASE; INSTICC, 2010, pp. 186–191.
[2] L. Aversano, M. Tortorella, Evaluating the quality of free/open source systems: a case study, in: Lecture Notes in Business Information Processing, Springer Verlag, 2011, pp. 119–134. 73(2).
[3] L. Aversano, M. Tortorella, Applying EFFORT for evaluating CRM open source systems, in: 12th International Conference on Product-Focused Software Process Improvement, PROFES 2011, Lecture Notes in Business Information Processing, Springer Verlag, 2011, pp. 202–216.
[4] C.P. Ayala, D.S. Cruzes, X. Franch, R. Conradi, Towards improving OSS products selection – matching selectors and OSS communities perspectives, in: 7th International Conference on Open Source Systems, Springer, 2011, pp. 244–258.
[5] V.R. Basili, G. Caldiera, H.D. Rombach, Goal question metric approach, in: J.J. Marciniak (Ed.), Encyclopedia of Software Engineering, Wiley Interscience, 1994, pp. 528–532.
[6] J. Bansiya, C.G. Davis, A hierarchical model for object-oriented design quality assessment, IEEE Transactions on Software Engineering 28 (1) (2002) 4–17.
[7] E.W.N. Bernroider, V. Stix, Profile distance method: a multi-attribute decision making approach for information system investments, Decision Support Systems 42 (2) (2006) 988–998.
[8] B. Birdogan, C. Kemal, Determining the ERP package-selecting criteria: the case of Turkish manufacturing companies, Business Process Management Journal 11 (1) (2005) 75–86.
[9] B.W. Boehm, J.R. Brown, M. Lipow, Quantitative evaluation of software quality, in: 2nd International Conference on Software Engineering, IEEE Comp. Soc. Press, Los Alamitos, 1976, pp. 592–605.
[10] B.W. Boehm, J.R. Brown, H. Kaspar, M. Lipow, G. McLeod, M. Merritt, Characteristics of Software Quality, Elsevier North Holland Pub., Co., 1978.
[11] S. Bueno, J.L. Salmeron, Fuzzy modeling enterprise resource planning tool selection, Computer Standards & Interfaces 30 (3) (2008) 137–147.
[12] A. Cau, G. Concas, M. Marchesi, D.M. German, G. Robles, Method for qualification and selection of open source software (QSOS) GNU free documentation license, Mendley 42 (April) (2006) 1–6.
[13] S.R. Chidamber, C.F. Kemerer, Towards a metrics suite for object-oriented design, in: International Conference of Object-Oriented Programming Systems, Languages, and Applications – OOPSLA, ACM, New York, 1991, pp. 197–211.
[14] J.P. Confino, P.A. Laplante, An open source software evaluation model, International Journal of Strategic Information Technology and Applications (IJSITA), IGI Global 1 (1) (2010) 60–77.
[15] V. Del Bianco, L. Lavazza, S. Morasca, D. Taibi, A survey on open source software trustworthiness, IEEE Software 28 (5) (2011) 67–75.
[16] V. Del Bianco, L. Lavazza, S. Morasca, D., Taibi, D. Tosi, The QualiSPo approach to OSS product quality evaluation, in: 3rd International Workshop on Emerging Trends in Free/Libre/Open Source Software Research and Development, ACM, New York, 2010, pp. 23–28.
[17] V. Del Bianco, L. Lavazza, S. Morasca, D. Taibi, D. Tosi, An investigation of the users' perception of OSS quality, in: 6th International Conference on Open Source Systems, Springer Verlag, 2010, pp. 15–28.
[18] W.H. DeLone, E.R. McLean, The DeLone and McLean model of information systems success: a ten-year update, Journal of Management Information Systems 19 (4) (2003) 9–30.
[19] J.C. Deprez, S. Alexandre, Comparing assessment methodologies for free/open source software: OpenBRR and QSOS, in: 9th International Conference on Product-Focused Software Process Improvement (PROFES'08), Springer Verlag, 2008, pp. 189–203.
[20] C. Ebert, Open source drives innovation, IEEE Software 24 (3) (2007) 105–109.
[21] C. Ebert, Guest Editor's introduction: how open source tools can benefit industry, IEEE Software Journals 26 (2) (2009) 50–51.

[22] N.E. Fenton, S.L. Pfleeger, Software Metrics: A Rigorous and Practical Approach, PWS Publishing Co., Boston, MA, 1998.
[23] F. Fui-Hoon Nah, Enterprise Resource Planning Solutions and Management, Idea Group Inc (IGI), 2002.
[24] A. Ghapanchi, M.H. Jafarzadeh, M.H. Khakbaz, Fuzzy-data envelopment analysis approach to enterprise resource planning system analysis and selection, International Journal of Information Systems and Change Management 3 (2) (2008) 157–170.
[25] A. Ghapanchi, M.H. Jafarzadeh, M.H. Khakbaz, An application of data envelopment analysis (dea) for erp system selection: case of a petrochemical company, in: ICIS 2008 Proceedings, 2008.
[26] B. Golden, Making open source ready for the enterprise: the open source maturity model, in: B. Golden (Ed.), Extracted From Succeeding with Open Source, Addison-Wesley Publishing Company, 2005.
[27] Ø. Hauge, C.P. Ayala, R. Conradi, Adoption of open source software in software-intensive organizations – a systematic literature review, Information & Software Technology 52 (11) (2010) 1133–1154.
[28] K. Hyoseob, C. Boldyreff, Open source ERP for SMEs, in: 3rd International Conference on Manufacturing Research, Cranfield University, 2005.
[29] M. Igbaria, End-user computing effectiveness: a structural equation model, OMEGA, The International Journal of Management Science 18 (6) (1990) 637–652.
[30] International Organization for Standardization, ISO 8402: Quality Management and Quality Assurance – Vocabulary International Organization for Standardization, ISO/IEC, 1994.
[31] International Organization for Standardization, ISO standard 9126: Software Engineering – Product Quality, part 1–4. ISO/IEC, 2001–2004.
[32] International Organization for Standardization, ISO standard ISO/IEC 25000:2005, Software Engineering – Software product Quality Requirements and Evaluation (SQuaRE), 2005.
[33] F. Kamseu, N. Habra, Adoption of open source software: Is it the matter of quality?, in: PReCISE, Computer Science Faculty, University of Namur, 2009
[34] S.H. Kan, V.R. Basili, L.N. Shapiro, Software quality: an overview from the perspective of total quality management, IBM Systems Journal 33 (1) (1994) 4–19.
[35] S.H. Kan, Metrics and Models in Software Quality Engineering, Addison-Wesley Professional, 2002.
[36] X. Liao, Y. Li, B. Lu, A model for selecting an ERP system based on linguistic information processing, Information Systems 32 (2007) 1005–1017.
[37] J.A. McCall, P.K. Richards, G.F. Walters, Factors in Software Quality, Nat'l Tech. Information Service vols. 1–3 (1977).
[38] P. Meirelles, C. Santos, J. Miranda, F. Kon, A. Terceiro, C. Chavez, A study of the relationships between source code metrics and attractiveness in free software projects, in: Brazilian Symposium on Software Engineering, IEEE Computer Society Press, Los Alamitos, 2010, pp.11–20.
[39] D. Nagy, A.M. Yassin, A. Bhattacherjee, Organizational adoption of open source software: barriers and remedies, Communications of the ACM 53 (3) (2010) 148–151.
[40] Open Source ERp Guru, Evaluation Criteria for Open Source ERP, 2008 <http://opensourceerpguru.com/2008/01/08/10-evaluation-criteria-for-open-source-erp/> (14.05.12).
[41] OpenBRR, Business Readiness Rating for Open Source, Intel, 2005 <http://docencia.etsit.urjc.es/moodle/file.php/125/OpenBRR_Whitepaper.pdf> (14.05.12).
[42] D. Reuther, G. Chattopadhyay, Critical factors for enterprise resources planning system selection and implementation projects within small to medium enterprises, in: IEEE International Engineering Management Conference, IEEE Comp. Soc. Press, Los Alamitos, 2004, pp. 851–855.
[43] M. Rofi, B. Zirawani, N.M. Salihin, H. Habibollah, Critical factors in ensuring the success of implementing open source ERP: case study in Malaysian small medium enterprise, in: Asia Pacific Industrial Engineering & Management Systems (APIEMS), 2008, pp. 849–857.
[44] G. Samoladas, D. Gousios, D. Spinellis, I. Stamelos, The SQO-OSS quality model: measurement based open source software evaluation, in: IFIP 20th World Computer Congress, Working Group 2.3 on Open Source Software, OSS 2008, Springer Verlag, 2008, pp. 237–248.
[45] C. Santos, J.M. Pearson, F. Kon, Attractiveness of free and open source software projects, in: 18th European Conference on Information Systems (ECIS), AIS Electonic Library, 2010, pp. 15–27.
[46] D. Spinellis, G. Gousios, V. Karakoidas, P. Louridas, P.J. Adams, I. Samoladas, I. Stamelos, Evaluating the quality of open source software, Electronic Notes in Theoretical Computer Science 233 (2009) 5–28.
[47] A. Srinivasan, Alternative measures of system effectiveness: associations and implications, MIS Quarterly 9 (3) (1985) 243–253.
[48] W.J. Sung, J.H. Kim, S.Y. Rhew, A quality model for open source selection, in: 6th International Conference on Advanced Language Processing and Web Information Technology, IEEE Comp. Soc. Press, Los Alamitos, 2007, pp. 515–519.
[49] R.T. Watson, L.F. Pitt, C.B. Kavan, Measuring information systems service quality: lessons from two longitudinal case studies, MIS Quarterly 22 (1) (1998) 61–79.
[50] D.A. Wheeler, How to evaluate open source software/free software (OSS/FS) programs, 2011 <http://www.dwheeler.com/oss_fs_eval.html> (14.05.12).
[51] C.C. Wei, C.F. Chien, M.J.J. Wang, An AHP-based approach to ERP system selection, International Journal of Production Economics (2005) 47–62 (96)1.
[52] B. Zirawani, M.N. Salihin, H. Habibollah, Critical factors to ensure the successful of OS-ERP implementation based on technical requirement point of view, in: 3rd Asia International Conference on Modelling & Simulation, IEEE Comp. Soc. press, Los Alamitos, 2009, pp. 419–424.